UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
11/775,617	07/10/2007	Ramesh Pokala	019287-0355575	5317

105727 7590 04/19/2017

Pillsbury Winthrop Shaw Pittman LLP (CA, Inc.)
PO Box 10500
McLean, VA 22102

EXAMINER
FABER, DAVID

ART UNIT	PAPER NUMBER
2177	

NOTIFICATION DATE	DELIVERY MODE
04/19/2017	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

Docket_IP@pillsburylaw.com

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Ex parte RAMESH POKALA and PRASAD PILLALA

Appeal 2015-005850
Application 11/775,617[1]
Technology Center 2100

Before ALLEN R. MacDONALD, HUNG H. BUI, and NABEEL U. KHAN, *Administrative Patent Judges*.

BUI, *Administrative Patent Judge*.

DECISION ON APPEAL

Appellants seek our review under 35 U.S.C. § 134(a) of the Examiner's Final Rejection of claims 1–19. We have jurisdiction under 35 U.S.C. § 6(b).

We AFFIRM.[2]

---

[1] According to Appellants, the real party in interest is CA, Inc. App. Br. 2.

[2] Our Decision refers to Appellants' Appeal Brief filed January 14, 2015 ("App. Br."); Reply Brief filed May 13, 2015 ("Reply Br."); Examiner's Answer mailed March 13, 2015 ("Ans."); Final Office Action mailed August 14, 2014 ("Final Act."); and original Specification filed July 10, 2007 ("Spec.").

STATEMENT OF THE CASE

Conventional client-server systems typically perform "server-side" input validation for web applications that communicate between a client and a server. Spec. ¶ 4. However, such a "server-side" validation can (i) introduce significant communication bottlenecks by transferring excessive amounts of data over a network and (ii) potentially degrade server response time, utilization, or other aspects of the server's performance. Spec. ¶ 4.

One way in which existing systems attempt to add "client-side" validation to web applications includes developing custom validation scripts to perform the "client-side" validation. However, a developer has to manually develop custom validation scripts (e.g., Javascript) and plug the scripts in at appropriate HyperText Markup Language (HTML) elements. The developer must also know an HTML element structure of an input component in order to properly plug in the validation script. As such, custom scripts for perform client-side input validation can often be a tedious and error-prone process. Spec. ¶ 5.

Appellants' invention seeks to provide a generic framework for performing client-side validation by incorporating validation scripts (e.g., Javascript) for various input types, without requiring a developer to manually develop validation scripts (e.g., JavaScript) or account for how and/or where to plug-in the validation script for each component. Spec. ¶ 7. For example, a developer may (i) create a web application including one or more input components, and (ii) enable client-side input validation for these components by implementing a validation script associated with the generic framework in advance and by indicating which validation script to use for the client-side validation. Spec. ¶ 7.

Claims 1, 10, and 19 are independent. Claim 1 is illustrative of Appellants' invention, as reproduced below with disputed limitations in italics:

> 1. A method to perform client-side input validation in a client-server environment, the method comprising:
>
> *receiving, at a client computing device, a web page from a server computing device, wherein the web page comprises an input component and a validator tag associated with the input component and wherein the validator tag in the web page received at the client computing device comprises an attribute value that constrains a valid input definition associated with the input component in the web page received at the client computing device;*
>
> *executing a renderer locally stored on the client computing device to locally process the input component and the validator tag associated with the input component, wherein the renderer is locally executed on the client computing device to locally generate, based on at least the validator tag of the received web page, source code to create a component object corresponding to the input component and locally validate whether an input associated with the component object satisfies the valid input definition associated with the validator tag*; and
>
> displaying the web page on the client computing device, wherein the displayed web page references a script file stored on the client computing device containing function that locally executes the source code on the client computing device to perform client-side validation on the input associated with the component object.

App. Br. 26 (Claims App.).

Evidence Considered

Scholz et al.	US 2003/0078949 A1	Apr. 24, 2003 ("Scholz")
Kougiouris et al.	US 2004/0039993 A1	Feb. 26, 2004 ("Kougiouris")

Dziejma	US 2005/0028084 A1	Feb. 3, 2005
Smith et al.	US 7,117,504 B2	Oct. 3, 2006 ("Smith")
Weinberg et al.	US 7,734,625 B2	June 8, 2010 ("Weinberg")

Felgall, Felgall Javascript - Validation on Submit, http://classic-web.archive.org/web/20040324050329/http://www.felgall.com/jstip27.htm

Examiner's Rejections

(1) Claims 1–4, 10–13, and 19 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Dziejma and Scholz. Final Act. 3–7.

(2) Claims 5 and 14 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Dziejma, Scholz, and Smith. Final Act. 7.

(3) Claims 6, 9, 15, and 18 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Dziejma, Scholz, Smith, and Felgall. Final Act. 7–9.

(4) Claims 7 and 16 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Dziejma, Scholz, Smith, Felgall, and Kougiouris. Final Act. 9–10.

(5) Claims 8 and 17 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Dziejma, Scholz, Smith, Felgall, and Weinberg. Final Act. 10–11.

Issues on Appeal

Based on Appellants' arguments, the dispositive issues on appeal are (1) whether the cited prior art teaches or suggests several limitations of independent claims 1, 10, and 19; and (2) whether the Examiner has

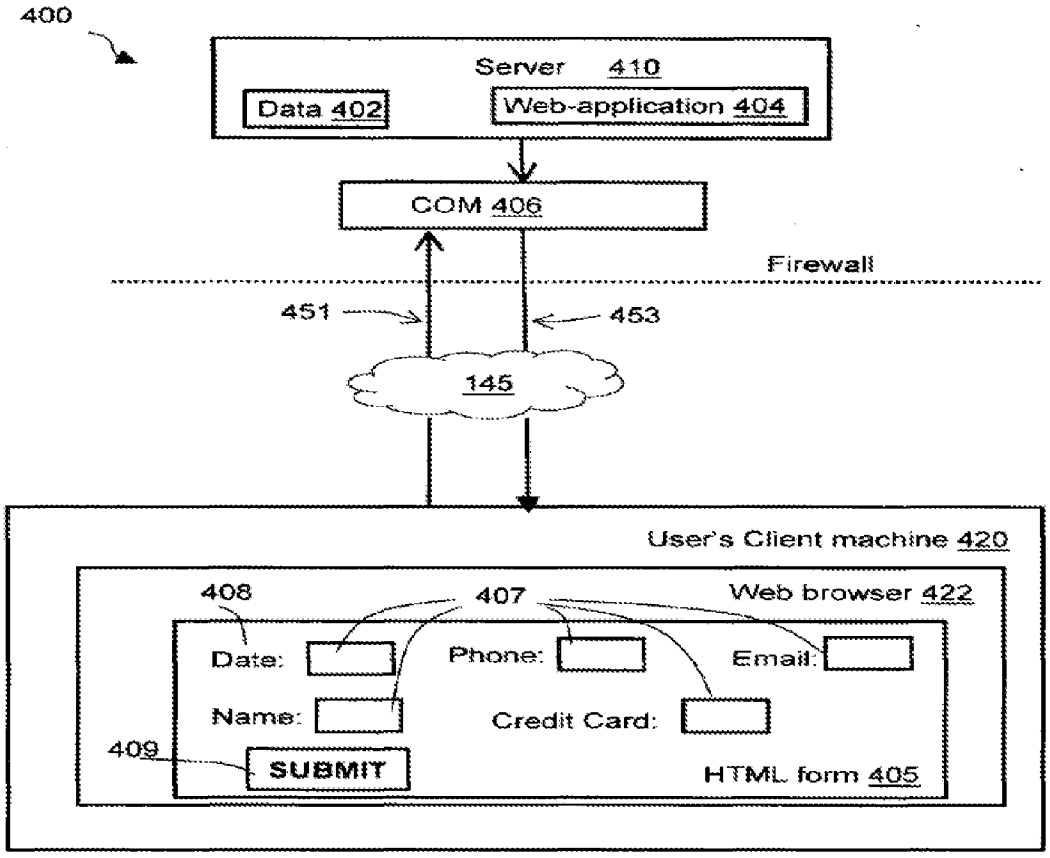
articulated reasoning with rationale underpinning to combine the prior art references. App. Br. 6–16; Reply Br. 2–6.

ANALYSIS

35 U.S.C. § 103(a): Claims 1–4, 10–13, and 19 based on Dziejma and Scholz

The Examiner finds Dziejma and Scholz teach all limitations of independent claims 1, 10 and 19. Final Act. 3–6. For example, the Examiner finds Dziejma teaches Appellants' claimed "method to perform client-side input validation in a client-server environment" including: (1) "receiving, at a client computing device, the web page from a server computing device, wherein the web page comprises an input component and a validator tag associated with the input component . . ." in the form of HTML form 405 downloaded from server 410, shown in Figure 1, including input fields 407–409 and embedded markers as validator tags (Final Act. 3 (citing Dziejma ¶¶ 9–10, 23–26, 41)); (2) "[e]xecuting a renderer locally stored on the client computing device to locally process the input component and the validator tag associated with the input component" in the form of HTML form 405 having fields validated or processed locally at client 420 (*id.* at 3–4 (citing Dziejma ¶¶ 10, 41)); and (3) "[d]isplaying the web page on the client computing device." (*Id.* at 4 (citing Dziejma ¶¶ 5, 9–10, 24–41)).

Dziejma's Figure 1 shows a client-server system to perform client-side input validation, as reproduced below:



Dziejma's Figure 1 shows a client-server system with client 420 arranged to receive web page 405 (e.g., HTML form) including input fields 407–409 and embedded markers as validator tags from server 410 to perform client-side input validation.

To support the conclusion of obviousness, the Examiner relies on Scholz for teaching the same "renderer [] . . . to locally generate, based on a validator tag of a web page, source code to create a component object corresponding to the input component and locally validate the input associated with the component object." *Id.* at 5 (citing Scholz ¶¶ 95–102, 106, 109, 116, 148, 150, 175).

Appellants dispute the Examiner's factual findings regarding Dziejma and Scholz and the Examiner's rationale to combine Dziejma and Scholz. In particular, Appellants contend neither Dziejma nor Scholz teaches or

suggests the disputed limitations: (1) "receiving, at a client computing device, a web page from a server computing device, wherein the web page comprises an input component and a validator tag associated with the input component . . ." and (2)

> executing a renderer locally stored on the client computing device to locally process the input component and the validator tag associated with the input component, wherein the renderer is locally executed on the client computing device to locally generate, based on at least the validator tag of the received web page, source code to create a component object corresponding to the input component . . .

as recited in claim 1, and similarly recited in claims 10 and 19. App. Br. 8. According to Appellants,

> there is no apparent reference in the cited portions of Dziejma and Scholz of generation of source code based on at least a validator tag of a received web page. For example, there is no apparent mention in the cited portions of Dziejma of source code generation based on something of a web page received at a client computing device. Indeed, the cited portions of Dziejma appear to be silent regarding source code generation generally, let alone at a client and let alone based on something of a web page received at a client computing device. Similarly, the cited portions of Scholz also fail to source code generation based on something of a web page received at a client computing device. For example, there is no apparent mention of receiving a web page comprising a validator tag as claimed at a client device in the cited portions of Scholz, let alone generating source based on at least such a validator tag and let alone generating such source code at a client computing device.

App. Br. 8 (emphasis added).

Appellants acknowledge paragraph 174 of Scholz "refers to where the validation of input occurs, i.e., where the validation code generated in

Scholz is executed — client-side or server-side," but argues Scholz "is silent about source [code] generation at a client device, let alone about source code generation based on something of a web page received at a client device" and "[t]here is no clear disclosure or teaching of generating anything on a client, let alone generating source code, based on at least a validator tag of a received web page, at the client." App. Br. 8–9, 11; Reply Br. 3–4. Likewise, Appellants argue

> "[t]he cited portions of Dziejma and Scholz both relate, at best, to systems that generate input validation code or functions (e.g., to validate inputs entered into a web page) on or at a server, rather than a client. While input validation may occur at a client in both the systems of Dziejma and Scholz, the cited portions of Dziejma and Scholz both disclose and teach that all the code to perform such validation is, at best, generated at the server side and provided to the client.

App. Br. 10.

Lastly, Appellants argue because "both Dziejma and Scholz relate to generating the input validation code at the server and providing that code to a client," there is no reason "to modify the system described in Dziejma to introduce client-side generation of source code therein, when neither the cited portions of Scholz or of Dziejma disclose or teach client-side generation of source code nor disclose or teach generation of source code based on something received at a client." App. Br. 12.

We do not find Appellants' arguments persuasive. Instead, we find the Examiner has provided a comprehensive response to Appellants' arguments supported by a preponderance of evidence. Ans. 4–11. As such, we adopt the Examiner's findings and explanations provided therein. *Id.* As recognized by the Examiner, Appellants' arguments are predicated upon:

(1) multiple attacks of Dziejma and Scholz individually when the rejection is based on a combination of references. *In re Keller*, 642 F.2d 413, 425 (CCPA 1981); and (2) mischaracterization of Dziejma and Scholz. App. Br. 8–16. For example, Dziejma is not relied upon for disclosing a renderer to generate source code, based on at least the validator tag of a web page, to create a component object corresponding to the input component and validate the input associated with the component object (Scholz is). Ans. 4, 6 (citing Scholz ¶¶ 95–102, 106, 109, 116, 148, 150, 175). Nor is Scholz relied upon for an express disclosure of "receiving, at a client computing device, a web page from a server computing device, wherein the web page comprises an input component and a validator tag associated with the input component" (Dziejma is). *Id.* at 5 (citing Dziejma ¶¶ 9–10, 41).

Contrary to Appellants' arguments, both Dziejma and Scholz teach a client-side input validation, and not server-side as alleged by Appellants. For example, Dziejma teaches the same (i) conventional client-server systems performing "server-side" input validation and (ii) custom validation script for "client-side" input validation as acknowledged by Appellants. *See* Dziejma ¶¶ 5–7; Spec. ¶¶ 4–5. Like Appellants' invention, to avoid the inefficiency of "server-side" input validation or custom programming, Dziejma proposes sending an HTML form (i.e., web page) along with a form validation engine including embedded markers as validator tags (validator code) to a client machine for performing "form validation on the client side." Dziejma ¶¶ 9–11, 21. Similar to Dziejma and Appellants' invention, Scholz also addresses the problem of custom programming. Scholz ¶ 4. As a solution, Scholz proposes sending an HTML form (i.e., web page) including validation code to a client machine for performing "form validation on the

client side." *See* Scholz ¶ 92 ("[t]he form itself includes the validation code and thus performs the validation at the client (referred to as client-side validation)), ¶ 174 ("[t]he automatic form generation with input validation described herein is predominately described with reference to client-side execution (e.g., client-side JavaScript code))." In addition, Scholz also teaches "[t]he generation of such validation code is well-known to those skilled in the art" and such "validation code" can be obtained from different sources, for example, "tag library" or other components to validate user inputs. Scholz ¶ 108.

For the reasons set forth above, Appellants have not persuaded us of Examiner error. Accordingly, we sustain the Examiner's obviousness rejection of independent claims 1, 10, and 19 and respective dependent claims 3 and 12 which Appellants do not argue separately. App. Br. 16.

Claim 2 depends from claim 1, and further recites: "wherein the source code to locally validate whether the input associated with the component object satisfies the valid input definition comprises a JavaScript statement generated from the attribute value that constrains the valid input definition." Similarly, claim 11 depends from claim 10 and recites the same limitation. *See* App. Br. 29 (Claims App.).

Appellants argue neither Dziejma nor Scholz teaches or suggests the recited "Javascript generated at the client as claimed." App. Br. 17. We disagree. As recognized by the Examiner, paragraphs 106–110, 114, and Table 3 of Scholz teach "executable code [] written in Javascript and includes the Javascript statement generated from the attribute value that constrains the valid input definition." Ans. 12.

Claim 4 depends from claim 1, and further recites: "wherein the renderer that is locally executed on the client computing device to locally generate source code further creates a validator representing the attribute value that constrains the valid input definition." Claim 13 depends from claim 10 and recites the same limitation. *See* App. Br. 29 (Claims App.).

Appellants argue "the cited portions of Scholz fail to disclose or teach . . . a validator object representing an attribute value that constrains a valid input definition, wherein the attribute value comes from a validator tag of a received web page as claimed." App. Br. 18. Again, we disagree. As recognized by the Examiner,

> Scholz discloses performing client side form generation (i.e. source code generation) In addition, Scholz generates validation code into the form (see 0108-0110). This validation code includes attribute values. (0110 discloses the example of an value being 32). Thus, since Scholz discloses the automatic form generation occurs on the client, and includes generating validation code, then Scholz discussion of a validator object/code being generated occurs on the client.

Ans. 13.

For these reasons, we also sustain the Examiner's obviousness rejection of dependent claims 4 and 13 based on Dziejma and Scholz.

With respect to remaining dependent claims 5–9 and 14–18, we remain unpersuaded by Appellants' arguments and adopt the Examiner's findings and explanations provided therein. Final Act. 7–11; Ans. 19–24.

CONCLUSION

On the record before us, we conclude Appellants have not demonstrated the Examiner erred in rejecting claims 1–19 under 35 U.S.C. § 103(a).

DECISION

As such, we AFFIRM the Examiner's Final Rejection of claims 1–19.

No time period for taking any subsequent action in connection with this appeal may be extended under 37 C.F.R. § 1.136(a)(1)(iv).

AFFIRMED